34556/JFO/B600

~~MEMORY STRUCTURE~~

CROSS-REFERENCE TO RELATED APPLICATIONS

This patent application claims the benefit of the filing date of United States Provisional Patent Application Serial No. 60/117,481, filed January 27, 1999, entitled ETHERNET SWITCHING and United States Provisional Patent Application Serial No. 60/127,147, filed March 31, 1999, entitled ETHERNET SWITCHING, the entire contents of both of which are expressly incorporated herein by reference.

BACKGROUND OF THE INVENTION

The invention herein relates to packet-based network switches; particularly to high-speed multiport packet-based network switches; and more particularly to memory structures, and associated operational techniques, for high-speed multiport packet based network switches.

Present-day throughput demands on packet-based switched networks have created the necessity for switches exhibiting increasingly higher performance. It is most desirable to achieve the transmission speed of the physical transport medium, i.e. to be close to "wire speed." For high-speed LAN protocols, including those collectively called "Fast Ethernet," switches typically associated with operations incorporating OSI Reference Model Layer 2 (Data Link Layer) and Layer 1(Physical Layer) are employed to meet the performance requirements reliably and economically. As the complexity of such devices increases, however, significant trade-offs, for example, between performance, scalability, and affordability may arise.

SUMMARY OF THE INVENTION

The present invention includes a shared memory structure which has an Address Resolution Table for resolving addresses in a packet-based network switch; and a Packet Storage Table that is adapted to receive a packet for storage in the packet-based network switch.

In another aspect of the invention, the Address Resolution Table is implemented using an associative memory structure, including, without limitation, a direct-mapped (one-way associative) memory. This memory may be searched for destination addresses using a destination address key direct-mapped address search. The shared memory structure also can include a Transmit Descriptor Table. It is desirable to have a Transmit Descriptor Table corresponding with a packet-based network transmit port.

Furthermore, the shared memory structure may include a Free Buffer Pool having multiple memory buffers, each having a pre-determined number of memory locations associated therewith. A bit-per-buffer technique can be used for tracking buffer status also can be used.

The invention can be practiced in a packet-based network switch which implements IEEE Standard 802.3 communication protocols. In the switch, the associated Address Resolution Table and the Packet Storage Table also employ a shared memory structure. The switch may have multiple ports and, indeed, it is contemplated that a switch, according to the present invention, have four, eight, nine, or more, such ports.

DESCRIPTION OF THE DRAWINGS

FIG. 1 is an illustration of a multi-port packet-based switch having an embodiment of the present invention;

FIG. 2 is an illustration of one memory block configuration having an embodiment of the shared memory structure according to the invention herein;

FIG. 3 is an illustration of plural memory blocks, each having the memory structure illustrated in FIG. 2;

FIG. 4 is an illustration of one embodiment of an ARL Address Table of the present invention;

FIG. 5 is a table illustrative of storage for an individual 66-byte packet in the context of the invention herein;

FIG. 6 is a an illustration of a packet data bit mapping table implementing an embodiment of the present invention;

FIG. 7 is an illustration of a transmit descriptor pointer address as implemented by an embodiment of the present invention;

FIG. 8 is a block diagram of an embodiment of the free buffer manager;

FIG. 9 is a state diagram illustrating the operation of one embodiment of the buffer control finite state machine in FIG. 8.

DETAILED DESCRIPTION OF THE INVENTION

The present invention is described in the context of the communication protocol defined by IEEE Standard 802.3, and supplemented, for example, by IEEE Standard 802.3u, which also is known as 100Base-T or "Fast Ethernet." Thus, embodiments of the present invention can be implemented in hybrid, or dual speed, 10/100BASE-T devices. One skilled in the art would realize that this contextual description is exemplary, and that the invention can be practiced in the context of other packet-based communication protocols, and at wire speeds surpassing those embodied, for example, by the 100BASE-T standard. Also, a skilled artisan would be familiar with the IEEE Standard 802.3, and thus would require no additional elaboration herein of these Standards to practice the invention. Furthermore, such IEEE Standards 802.3 are incorporated by reference herein in their entirety.

A packet-based Layer 2 switch typically includes fundamental components such as physical layer transceivers (PHY), media access controllers (MAC), an address management unit, a packet

switching fabric, random access memory, and the like. The physical transceiver layer, MAC, and other aspects of a packet-based switch can be defined by standards such as, for example, the IEEE 802.3 group of specifications (hereinafter referred to as "Ethernet" or, where appropriate, "Fast Ethernet"). The integration of some of these components is known in the art. However, total integration of all components onto a single chip may create performance trade-offs, depending, for example, upon the complexity of the switch. As the number of supported nodes increases, it becomes more difficult to meet power requirements and die size constraints, and still operate at, or near, wire speeds.

Among the functions supported, a Layer 2 switch resolves the destination of frames received at an ingress port by building a table of destination addresses and an associated egress port. An Ethernet destination address typically is a 48-bit value. Therefore, building a direct mapping for each possible address can require 2^{48} memory locations. Recognizing that only a small number of the 2^{48} addresses may be used in a LAN system, it is desirable to reduce the memory required to store the addresses, and to minimize the probability of an address search miss. Techniques to realize these goals include the use of a content-addressable memory (CAM), binary search algorithms, and hash tables with chain entries of depth greater than 1. However, such techniques can be costly to implement, and can degrade the frame rate resolution of destination addresses such that operation at wire speed can be difficult to maintain under some circumstances.

FIG. 1 illustrates a packet-based multiport switch 1 which includes an integrated switching controller 2, and a memory 3, external to switching controller 2. According to an embodiment of the present invention, it is contemplated that Packet Data Storage Table 4 be co-located with Address Resolution Table 5. In particular, it is most desirable that Packet Data Storage

Table 4 share memory with ARL Table 5. Further, memory 3 also can include Transmit Descriptor Table 6. Integrated switching controller 2 can include switching fabric 7, free buffer pool memory 8, free buffer pool memory manager 9, and MAC/PHY components 10a, 10b, and 10c.

An advantage of having a shared memory structure 3 as contemplated by the present invention is the reduction in device pin count and in system implementation costs. An advantage of implementing the invention as a direct-mapped address table is that the number of memory accesses required for address resolution per Ethernet frame can be about one cycle per Ethernet frame for address learning, and about one cycle per Ethernet frame for address resolution. Furthermore, the memory addressing logic required to access the ARL table can be minimized. It is desirable to use a direct-mapped/one-way associative address table, indexed by a key, for example, extracted from the thirteen least significant bits of the 48-bit Ethernet frame destination address field.

In one embodiment of the invention, ARL Table 5 may be used without a shared memory structure. In this case, it is desirable for the table to be configured as an one-way associative, i.e., direct mapped, memory. In embodiments of the invention in which the ARL Table 5 is shared with Table 4, Table, 6, or both, as well as with pool memory 8, it may be desirable to use another type of memory structure, including, without limitation, an ***n***-way associative memory, a hash table, binary searching structure, and a sequential searching structure. One skilled in the art could readily select the appropriate a search technique for a given structure.

By using the one-way associative memory configuration for ARL Table 5, address resolution can be made simple, and memory access expeditious, thereby reducing the switching bandwidth needed to determine the packet destination port address, and to

allow the Packet Data Storage Table 4 to be co-located with ARL Table 5. This direct-mapped configuration of ARL Table 5 reduces the switching bandwidth needed to determine the packet destination port address, and permits an associated device to operate at, or near, wire speed.. Also, the direct mapping can be a significant factor in implementing the single, shared memory structure for the ARL Table 5 and Packet Data Storage Table 4, which facilitates switch operation at wire speeds.

The implementation of shared memory 3 and the implementation of a direct-mapped ARL Table 5, alone and together, are more desirable techniques to increase bandwidth than merely increasing clock frequency because operations using faster clock frequencies typically result in increased power requirements, and a need for faster memory which, itself, can add to the cost, and complexity, of the associated product. Thus, where it is desired to contain device power requirements and to minimize switch cost, the aforementioned approaches are beneficial.

By using a preselected portion of the packet destination address as an index into ARL Table 5, a address match can be resolved quickly, and the packet passed to the appropriate port for transmission. This destination address key direct-mapped address search enables multiport packet-based switch 1 to be operable, for example, at wire speed in full-duplex, non-blocked, 100Base-TX operations. One skilled in the art would realize that the contemplated invention can be practiced in environments other than 100BASE-T environments and at wire speeds in excess of 100 Mb/s.

FIG. 2 provides an illustration of one embodiment of a memory map 100 that can implement a block of memory such as memory 3 in FIG. 1. At first address locations 11 (00-CF), there exists a single buffer 12 for an individual packet. A transmit descriptor table, similar to Transmit Descriptor Table 6 in FIG. 1, is created by allocating sufficient memory beginning at first

memory location 13 (D0) to second memory location 15 (D8) which encompass port 0 transmit descriptor 14 through port 8 transmit descriptor 16. Also, an address resolution table similar to ARL Table 5 in FIG. 1, can be created by allocating a memory segment 17 such that it contains a preselected number of ARL table entries 18 (e.g., 32 entries).

With one buffer per packet, only one transmit descriptor read per packet is performed, eliminating multiple memory accesses to find, for example, a linked list of buffers in an external memory. Given the starting address of the frame and the length of the frame in the transmit descriptor, only one access is executed in order to locate the entire packet to be transmitted. In a typical linked-list buffer approach, employing a small, fixed buffer block size, additional transmit descriptor reads may be required in order to locate each subsequent block. Each additional read signifies an undesirable reduction in available bandwidth.

Furthermore, the single buffer per packet approach as contemplated herein reduces the number of buffers that need to be searched. A skilled artisan would appreciate the significant bandwidth savings that can be attributed to the one buffer per packet approach. The single buffer-per-packet technique enhances the feasibility of the bit-per-buffer free buffer pool tracking technique, as well, and the need to search a large buffer pool structure can be mitigated or eliminated. In view of the foregoing, it can be seen how embodiments of the contemplated invention effect switch operation at, or near, wire speed.

FIG. 3 is illustrative of the scalability of this shared memory structure in that the memory structure described in FIG. 2 can be allocated in address range 19 of memory block 20. A skilled artisan would realize that one or more such blocks can be used to achieve the desired design criteria.

FIG. 4 illustrates one embodiment of the direct-mapped address table indexing using, for example, a 13-bit key derived from the 48-bit MAC address value, i.e., the Ethernet frame destination address field 21. As previously described, the least significant bit 23 of address value 21 is mapped to the least significant bit 24 of key 22. In this example, the address table entries, therefore, are offset in the address space from the index by $F0_h$. The most significant bit location 25 can obtain its value 26 from bit 35 of the corresponding MAC address value 21. If desired, a fourteenth bit from MAC address 21 can be used to provide a bit value 28 for the most significant bit 27 for key 22.

Thus, a packet-based switch implementing the shared memory structure according to the contemplated invention performs one memory read for address resolution, and one memory write for address learning, to the address table for each frame received. The reduced overhead provided by embodiments of present invention leads to a reduction in memory accesses per Ethernet frame (in this example, a frame is 64 bytes in length, and the associated bus width is 64 bits). The number of such memory accesses can be characterized as: one cycle per frame for address resolution; one cycle per frame for address learning; one cycle per frame for transmission read; one cycle per frame for transmission write; one cycle per eight bytes for a frame data read; and one cycle per eight bytes for a frame data write.

The single access for both read and write can be attributed to the single-entry direct-mapped address table. Using this configuration, each MAC address maps to a single location in the address table. Therefore, only one access may be needed to read or write the MAC address. A single-entry direct-mapped address table may increase the probability of address collisions. However, the probability of these collisions can be reduced by mapping over a larger number of MAC address bits, such as the 14

bits illustrated in FIG. 4. The single MAC address read and write for each Ethernet frame can contribute to the ability of switch 1, in FIG. 1, to operate at wire speed, in a full-duplex, non-blocking manner.

To further enhance the functionality of switch 1 in FIG. 1, a transmit descriptor request may be made during the transmission of a previous frame, thereby removing the transmit descriptor reads from the generation of latency. Also, it is desirable that a FIFO structure be used so that a first portion of the FIFO data can be read to initiate transmission while the remaining portion of the FIFO structure is still receiving data.

In one embodiment of the invention, memory structure 3 of Figure 1 employs a 64-bit memory bus operating with a 66 MHz system clock. Throughput can further be enhanced by implementing a memory arbitration scheme such as a weighted priority round-robin memory arbitration technique. This technique enhances the memory structure's quantization and prioritization of memory accesses, further reducing latency loss and bandwidth requirements.

An embodiment of the present invention contemplates the implementation of a memory arbiter that, in this example, provides arbitration for six types of memory accesses. The arbiter sets priority between the Ethernet ports as highest priority and that of an expansion port as the lowest priority for each of the memory access types. Each access type is also prioritized such that the access type meets the latency requirement for maintaining wire speed switching of the supported function. The selected arbitration and associated priority are as shown in Table 1.

TABLE 1

Access Type	Priority	Cycles/Access	Access/Frame #
Frame Data Writes	1	4	2
Frame Data Reads	2	4+2 Turnaround	2
Transmit Descriptor Read	3	1+2 Turnaround	1
Destination Address Read	4	1+2 Turnaround	1
Transmit Descriptor Write	5	1	1
Source Address Write	6	1	1
*64-byte Ethernet Frame			

The cycles/access number refers to the number of system clock cycles required to perform memory access when interfacing, for example, to an external synchronous static RAM in flowthrough mode, with 64-bit data word width.

Data packets can be stored in Packet Data Storage Table 4 of FIG. 1, with a packet data address portion, and a packet data value portion. References to packets are often passed within switch 1, which can, for example, use the upper nine buffer bits of Table 4 as a pointer value. These pointer values are passed between the Free Buffer Manager 9 and ports 10a, 10b, 10c. The data address pointer value can also be passed between the switch Rx ports and the switch Tx ports via the transmit descriptor, which is similar to descriptor 14.

FIG. 5 illustrates how packet data values can be stored. In the example of FIG. 5, a 66-byte packet is stored. As seen in FIG. 5, it is desired to store packet data in 64-bit wide memory segments, such that the efficiencies brought about by the 64-bit wide memory data path are further realized.

FIG. 6 is exemplary of mapping a transmission format to other selected memory formats. Although the format normally used to display Ethernet data is a byte-stream format 40, FIG. 6 also

displays the Ethernet data in a bit-stream format 41, a nibble-stream format 42, a byte-stream format 43 and a word-stream format 44.

In FIG. 1, it is desired that there be one Transmit Descriptor Table 6 for each transmit port. Thus, a switch having multiple ports (e.g., 4, 8, or 9 ports) could use a corresponding number of Transmit Descriptor Tables 6 (e.g., 4, 8, or 9 tables). It is also desired that each Transmit Descriptor Table 6 consist of a circular queue structure, i.e., a FIFO, that can hold a pointer value for each buffer in switch 1. Typically, a circular queue structure (FIFO) has a tail pointer and a head pointer that are maintained in each Tx block. When the values are the same, the queue is empty.

FIG. 7 illustrates the desired structure of a head pointer, a tail pointer, or both. In FIG. 7, head pointer 45 is described in this example, although a tail pointer can have the same structural format. Port ID 46 is desired to be static for each transmit port. Nine-bit pointer value 47, in this particular example, is indicative of the head pointer value. Where a reduced amount of memory is used to implement Table 6 of switch 1 in FIG. 1, then the sixteenth bit, 48, of FIG. 7 can be forced low on all memory accesses, having the effect of wrapping the transmit descriptor queues to fit within the available memory without affecting switch 1 operation.

FIG. 8 is an embodiment of a free buffer manager 50 similar to free buffer manager 9 shown in FIG. 1. Manager 50 can include a buffer free bus controller 51, a pipeline buffer search engine 52, a buffer control finite state machine 53, a buffer bus register 54 and a buffer grant bus controller 55. It is desirable that register 54 be a LIFO and, for the purposes of the description herein, register 54 is an eight-location LIFO. It is the responsibility of manager 50 to "grant" new buffers to ports before a data packet is received, and to collect, or

"free," used buffers from ports that have transmitted a packet. Typically, one grant operation will be needed for each packet received by the switch, and one free operation will be needed for each packet transmitted by the switch.

In an embodiment of the present invention, a fixed number of buffers are employed. Used buffers are those that have been granted to a receive port, but have not yet been returned, or freed, by a transmit port. All of the remaining buffers are designated "unused". It also is the buffer manager's responsibility also to track unused buffers so that they can be granted. Although one simple method to track unused buffers is to maintain a buffer list, such a list may create undesirable space limitations on a switch device because the list area must be long enough to store all of the buffers in a system and, further, each location in the list must be able to store the number of, or a pointer to, any buffer in the system. In a device having 512 list locations, for example, with each location having a corresponding nine-bit pointer, 4608 bits of storage would be required.

By contrast, another embodiment of the invention herein, implementing a bit-per-buffer method of tracking free buffers, reduces the storage requirement to only 512 bits, with each bit corresponding to a specific buffer. Here, a used buffer is indicated by setting the corresponding buffer bit. For example, setting the 368 bit in free buffer pool memory 8 in FIG. 1, can indicate that buffer 368 is currently being used.

Although this method does present an economy of storage and circuit area, it is further desired to employ a pipelined engine 52 to search for buffers in the bit array, such that the impact of "free" operations on search speeds is limited and that fast grant rates are allowed. Register 54 is preferred to be an eight-location LIFO to further increase the peak grant rate of search engine 52. Buffer free bus controller 51 captures

requests 58 for the freeing of buffers, and presents request 59 to search engine 52. In addition, controller 51 can provide a similar request 56 to finite state machine 53. Register 54 also provides a status signal 57 to finite state machine 53 and, in conjunction with request data signal 56 from free bus controller 51, buffer control finite state machine 53 can select one of a set of defined states.

The state diagram of FIG. 9 illustrates the three states of state machine 53 in FIG. 8. These three states can include:

1) SEARCH (61) - search for zero-valued bits that are in the buffer control array, indicating the location of a free buffer;

2) FREE (62) - write a zero to a bit location specified by free controller 51, thus freeing the associated buffer for allocation; and

3) ALLOCATE (63) - write a one value to a bit location that was identified during search state 61 by search engine 52.

Returning to FIG. 8. Buffer Search Engine 52 is preferred to be pipelined in both address and data paths, around the buffer control bit memory array, in order to expedite the identification of available buffers. The eight-location LIFO 54 can store the locations of allocated buffers until they are needed by the Buffer Grant Bus Controller 55. Finally, Buffer Grant Bus Controller 55 waits for requests 59 from the received port for buffers and presents the buffer location 60 if available from the LIFO.

The foregoing merely illustrates the principles of the invention, and it will thus be appreciated that those skilled in the art will be able to devise various alternative arrangements which, although not explicitly described herein, embody the principles of the invention within the spirit and scope of the following claims.